



SEAN ALLAN

The Missing Piece

First Principles Engineer & Creative Technologist

Background

I am a multidisciplinary education, technology, and community-engagement professional with more than a decade of experience working across classrooms, families, technical systems, and community needs.

The common thread in my work is accessibility. I am particularly effective at taking complicated systems, information, or problems and translating them into something people can understand and actually use.

In public education, that has meant helping families navigate IEPs and educational systems, coordinating multidisciplinary teams, supporting educators, responding to crises, and working extensively with high-needs and underserved communities. My responsibilities frequently grew beyond my formal role because administrators trusted me to step into difficult situations, identify what was not working, and help build a practical solution.

My technology and programming experience strengthened the same skill set from another direction: systems thinking, pattern recognition, troubleshooting, and communicating across people with very different areas of expertise.

Whether I am working with a family, educator, community organization, or technical team, my approach is similar: understand the barrier, make the complex understandable, and find a practical way forward

Four Years of Focused Progress

Approximately four years of concentrated, self-directed programming have produced applications, custom engines, local AI integrations, document tools, and rendering systems. Earlier technical support, classroom practice, and venture planning provide experience in diagnosing problems, defining requirements, and coordinating people. The work demonstrates rapid progression; individual projects establish the scope of each capability.

READING MAP

01 Profile / 02 Architecture & Graphics / 03 AI & Tooling / 04 Skills & Advantages / 05 Services & Principles



Software Architecture & Engines

I design how interfaces, services, processing engines, stored information, and local computing resources operate together. Defined inputs, outputs, and ownership boundaries give each component a clear place in the application.

My mathematics engines use deterministic algorithms, typed problem structures, configurable difficulty, and expected answer evaluation. Explicit rules support behavior that needs a defined correctness criterion; local AI supports generation and interpretation where appropriate.

The website demonstrates the same ownership approach in React: pages compose content, visual systems own their rendering layers, and local controllers coordinate interaction and cleanup.

Graphics & Interaction Engineering

I build interactive systems using Canvas, WebGL, GLSL, SVG, and CSS. Work includes procedural particles, material composition, holographic portraits, custom typography, and coordinated scene transitions.

Asset readiness and explicit phases determine when effects reveal, respond, and exit. Visibility tracking, rendering budgets, and disposal paths govern optional rendering work. Masks and textures generated with Python connect asset preparation with runtime composition.

Development workbenches expose the real components for isolated adjustment and preset export. These controls manage rendering work; performance depends on the device and the combined effects in use.

How I Approach the Work

I start with mathematical rules and physical constraints, then build the engines and architecture that express them. Owning the underlying logic lets me target specific problems precisely across education, graphics, document processing, and computing systems.

I learn through simultaneous study and application. Mathematics describes the relationships I observe; code turns them into models I can experiment with and refine.



Applied AI, Data & Delivery

I build application-facing services around locally hosted language models. TypeScript, Node.js, Express, and HTTP APIs connect requests, model inference, structured results, and application interfaces. The educational Alpha combines local generation with instructional data and conventional algorithms.

Document processing work uses Python, Tesseract, Poppler, and PowerShell to extract and stage information from PDFs and scanned material. Structured output and source references support review of the processing results. Separate local retrieval workflows keep reference collections apart from source records, carry document identifiers into evidence packets, and flag unsupported claims or open questions for human checking.

Connection-aware startup is another delivery problem: browser hints and a timed transfer probe guide presentation tiers, while readable HTML provides a fallback. These policies estimate delivery conditions rather than detecting physical signal strength. The educational system remains an Alpha; deployment requirements are assessed separately.

Automation, Tooling & Infrastructure

I develop scripts for staging data, running batch processes, preparing caches, packaging outputs, and reviewing operations. Python/Pillow handles asset preparation; PowerShell and TypeScript coordinate processing and development tasks.

I also compare cloud and local workload costs and estimated emissions using run volume, hardware life, task and idle power, facility overhead, and embodied emissions. Scenario values are synthetic; a real recommendation requires measured inputs and equivalent service quality.

My computing experience includes workstation assembly, hardware configuration, local model runtimes, and troubleshooting. Multimedia work includes OBS and VoiceMeeter audio routing, source isolation, filtering, and application-to-device signal paths. Earlier agricultural modeling, embedded automation, and haptic concepts remain historical research or prototypes with project-specific scope.



Skills & Advantages

Strengths

- **Self-Taught:** Independent research and implementation build depth in modern development techniques. Unfamiliar legacy conventions and dependencies require additional study before implementation
- **Modern Languages:** I use TypeScript and JavaScript for applications, and Python and PowerShell for processing and development tools. Database integration is historical experience
- **Deep System Knowledge:** I define responsibilities across interfaces, services, engines, and stored information. Mathematics engines use explicit rules and expected answers
- **Backend Integration:** I build application services connecting interfaces, local models, and structured document processing, with defined responsibilities for each stage
- **Inventive Problem Solver:** I translate specific requirements into custom logic, interfaces, or architecture and refine the behavior through experiments
- **Computing & Systems Integration:** I integrate workstation hardware, local model runtimes, and multimedia routing. Work includes GPU and storage configuration, troubleshooting, and OBS/VoiceMeeter signal paths; earlier physical interface work is experimental
- **Political Science & Systems Theory:** My political science background connects political theory, institutional analysis, and relationships within complex systems. I apply that perspective when examining incentives and how changes affect the whole

Limitations I Manage

- **Terminology Gap:** Independent learning can produce implementation knowledge before conventional vocabulary. Establishing shared terminology can require extra explanation
- **Legacy Stack Gap:** Modern stack experience is my strongest area. Unfamiliar legacy frameworks and operating conventions need additional research and compatibility assessment
- **Over-Polishing:** Refinement can continue beyond the immediate requirement. Clear acceptance criteria and milestones are needed to keep delivery priorities in control

How the Combination Helps

- **Modern Techniques: Granular Compatibility:** Modern languages and techniques meet deliberate compatibility assessment. Map the older system's behavior, isolate its constraints and introduce focused changes rather than assuming a rewrite
- **Stable Systems: Scoped Refinement:** System knowledge and explicit responsibilities guide failure isolation, predictable data flow and recovery. Acceptance criteria keep refinement focused on behavior the system actually needs
- **Original Solutions: Shared Technical Language:** Independent invention becomes easier to evaluate and integrate when the implementation is connected to shared terminology and explicit contracts. The result is a novel component others can understand and maintain



Services

Engineering & Consulting

I build prototypes, APIs, data pipelines, and software engines, then help connect them to the systems your organization already uses.

Creative Design & Web Pages

I design and code custom websites, animation, interactive demonstrations, and managed website rentals.

Working Together

I work directly with you on requirements, design and implementation. Each proposal defines deliverables, ownership, external costs, operating responsibilities and support. Original product invention is reserved for nonprofit projects. Current offers and pricing are published on the website.

Principles

- **Work Close to Home:** Run suitable processing on local hardware to reduce cloud dependence and repeated data transfer. Energy use still depends on the device, workload, and power source
- **Open Tools, Clear Choices:** Open-source software provides source code under a license that permits inspection and grants defined rights to use, change, and share it. I check each project's terms
- **Use Only What the Job Needs:** Choose focused processes, reusable tools, and computing resources sized for the task to avoid unnecessary cloud requests, large runtimes, and data movement
- **Accessible by Design:** Design around real needs, make complex behavior understandable, and provide accessible ways to operate the system

Contact & Public Work

contact@fullmetallogic.com

www.fullmetallogic.com

[Public GitHub portfolio](#)

English offline summary, adapted from the Complex reading depth. The live site and HTML Lite provide fuller detail and source links. Historical work, prototypes and planned features retain the qualifications stated here. This document is not a proposal or a promise of production performance.